เรื่องน่าอ่าน
ดร.ปัญญา เปรมปรีดี

เน็ตเวอร์กกิง
กับเรื่องของ ไคล์เอ็นต์เซอร์ฟเวอร์

ค้นหาสาเหตุของปัญหาการขาดประสบการณ์ด้านไคล์เอ็นต์เซอร์ฟเวอร์ พร้อมกับเสนอแนะแนวทางในการแก้ไข

ถ้าเราไปถามนักคอมพิวเตอร์ของไทยว่ามีประสบการณ์ด้าน "เน็ตเวอร์กกิง" (Networking) บ้างไหม เราจะได้รับคำตอบว่า "Yes" เป็นจำนวนมาก แต่ถ้าเราถามคนกลุ่มเดียวกันว่ามีประสบการณ์ในเรื่อง "ไคล์เอ็นต์เซอร์ฟเวอร์บ้างหรือไม่" เราจะได้รับคำตอบว่า "No" เกือบทั้งหมด เรื่องนี้นับเป็นเรื่องแปลก ทั้งนี้เพราะเมื่อไหร่ก็ตามที่เราเชื่อมโยงเครื่องคอมพิวเตอร์เข้าหากัน เราก็ต้องมีการติดต่อกันระหว่างโปรแกรมบนเครื่องเหล่านั้น และมันก็จะหนีไม่พ้นที่จะเกิดสถานการณ์เป็นไคล์เอ็นต์และเป็นเซอร์ฟเวอร์ ดังนั้นในสายตาของผมแล้วเรื่องเน็ตเวอร์กกิงกับเรื่องไคล์เอ็นต์เซอร์ฟเวอร์จึงเป็นเรื่องเดียวกัน และเราควรจะคิดไปพร้อม ๆ กัน ซึ่งนี่คือที่มาของหัวเรื่อง

สำหรับวันนี้ผมคิดว่าจะมาลองค้นหาสาเหตุของการขาดประสบการณ์ด้านไคล์เอ็นต์เซอร์ฟเวอร์ และจะลองเสนอเกร็ดความรู้ในการใช้ประโยชน์จากเทคนิคไคล์เอ็นต์เซอร์ฟเวอร์สักอย่างหรือ 2 อย่าง

สาเหตุของปัญหา

สาเหตุใหญ่ที่คนไทยเราไม่มีประสบการณ์ในเรื่องไคล์เอ็นต์เซอร์ฟเวอร์ก็เพราะระบบเน็ตเวอร์กที่เราซื้อมาใช้กันนั้นฝรั่งเขาทำด้านเซอร์ฟเวอร์มาให้เราแล้ว ยกตัวอย่างเช่น ในระบบงานที่ใช้พีซีแลน

พีซีแลนเป็นเน็ตเวอร์กของไมโครคอมพิวเตอร์ เจ้าไฟล์เซอร์ฟเวอร์นั่นคือเซอร์ฟเวอร์ ส่วนโปรแกรมที่อยู่ที่เวอร์กสเตชันต่าง ๆ นั้นเป็นไคล์เอ็นต์ แต่ผู้ผลิตเขาจัดการเขียนโปรแกรมทางด้านเซอร์ฟเวอร์ให้เราจนหมดแล้ว เราจึงเห็นแต่ทางด้านไคล์เอ็นต์ ซึ่งเหมือนกับการเขียนโปรแกรมในพีซีธรรมดา

อันที่จริงแล้วฝรั่งนี่ฉลาดคิด คือเขาเขียนโปรแกรมทางด้านเซอร์ฟเวอร์ไว้เป็นอย่างดี ดีจนสามารถหลอกให้เรามองเห็นว่าดิสก์ที่อยู่ที่ไฟล์เซอร์ฟเวอร์นั้นเป็นดิสก์ประเภท Remote การเรียกใช้ดิสก์ในไฟล์เซอร์ฟเวอร์จึงไม่แตกต่างจากการเรียกใช้ดิสก์ในเวอร์กสเตชันนั่นเอง และเนื่องจากดิสก์ในไฟล์เซอร์ฟเวอร์นี้มีขนาดใหญ่และง่ายต่อการใช้ร่วมกัน

เราก็เลยมักจะใช้ดิสค์ในไฟล์เซอร์ฟเวอร์เป็นที่เก็บข้อมูลทั้งหมด เจ้าดิสค์ที่เวอร์กสเตชันเลยไม่มีคนใช้ และถ้าจะให้ใช้ก็มักจะคิดไม่ออกว่าจะใช้ได้อย่างไร แบงค์ชาติมีพีซีแลนกว่า 30 วง แต่เกือบทั้งหมดไม่ใช้ดิสค์ที่เวอร์กสเตชันเลย ซึ่งเป็นสิ่งที่น่าเสียดายอย่างยิ่ง เวอร์กสเตชันแต่ละตัวนั้นมีฮาร์ดดิสค์ขนาด 40 ถึง 120 เมกะไบต์ เมื่อนับรวมกันวงละ 20 ตัวก็เป็นเนื้อที่กว่า 2 จิกะไบต์ ซึ่งใหญ่กว่าตัวไฟล์เซอร์ฟเวอร์เสียอีก

แนวทางแก้ไข

ครับ วันนี้ผมอยากจะเริ่มต้นที่จุดนี้แหละ

ดิสค์ในเวอร์กสเตชันเป็นของไคล์เอ็นต์ มันถูกควบคุมโดยโปรแกรมในเวอร์กสเตชันได้ ดังนั้นไคล์เอ็นต์ควรจะใช้ประโยชน์ให้เต็มที่ ปัญหาคือเราไม่รู้ว่ามีข้อมูลอะไรที่ควรเก็บไว้ทางด้านนี้บ้าง

มีครับ มีเยอะแยะเลย ผมจะขอยกให้ดู 1 กลุ่มก่อน กลุ่มนี้ผมเรียกว่า "Conversion Table" นี่คือตารางที่เราใช้แปลรหัส ยกตัวอย่างเช่น ในระบบการทำงาน พนักงานเราก็ต้องมีตารางรหัสหน่วยงาน ตารางรหัสชื่อตำแหน่ง ตารางรหัสชื่อสาขา ตารางรหัสชื่อจังหวัดและอำเภอ ฯลฯ ตารางเหล่านี้มีการเปลี่ยนแปลงไม่บ่อยนัก แต่ในโปรแกรมของเราต้องเรียกใช้อยู่บ่อย ๆ เช่น เมื่ออ่านเอาเรคอร์ดของนาย ก. ขึ้นมาแล้ว และกำลังจะแสดงเป็นรายละเอียดบนจอภาพ อย่างนี้เราก็ต้องเอารหัสหน่วยงานที่นาย ก. สังกัดอยู่นั้นไปค้นในตารางรหัสหน่วยงาน เมื่อได้ชื่อหน่วยงานมาแล้วก็เอาไปแสดงบนจอภาพ ต่อจากนั้นก็เอารหัสตำแหน่งไปค้นในตารางรหัสตำแหน่ง แล้วก็เอาชื่อตำแหน่งที่เราอ่านได้มาแสดงบนจอภาพ ฯลฯ ตารางแปลรหัสเหล่านี้เรามักจะออกแบบให้เก็บอยู่ในไฟล์เซอร์ฟเวอร์ ทั้งนี้เพราะโปรแกรมในเวอร์กสเตชันต่าง ๆ ต้องใช้งานร่วมกัน

เรื่องนี้ผมไม่เถียง แต่ผมเสนอแนะว่าโปรแกรมของเราควรก๊อบปี้มาลงที่เวอร์กสเตชัน แล้วใช้ไฟล์ที่เวอร์กสเตชันนี้ในการแปลรหัสต่าง ๆ ด้วยวิธีนี้เราสามารถลดภาระของไฟล์เซอร์ฟเวอร์ลงไปได้แยะเลย การก๊อบปี้ตารางต่าง ๆ ในตอนต้นของโปรแกรมเพียงครั้งเดียวนี้จะทำงานได้เร็วมาก ทั้งนี้พราะเราอ่านแบบ "ซีเควนเชียล" เจ้าไฟล์เซอร์ฟเวอร์นั้นโดยปกติจะมีแคชเมมโมรีเพื่อช่วยในการอ่านและบันทึกจานแม่เหล็ก ตารางรหัสขนาด 32 กิโลไบต์นั้นจะใช้เวลาอ่านจริง (Physical Read) เพียงครั้งเดียว การก๊อบปี้ตาราง 3-4 ตารางจึงเสร็จได้ในเสี้ยววินาที หลังจากนั้นแล้วเรามาใช้ไฟล์ในเวอร์กสเตชัน ซึ่งต่างคนต่างใช้ มันจึงไม่รบกวนกัน หลังจากที่เราทำงานจนเสร็จแล้ว โปรแกรมของเราก็อาจสั่งลบไฟล์เหล่านี้ทิ้งไปได้ ด้วยวิธีนี้เราจะไม่มีการเก็บข้อมูลซ้ำซ้อน ไฟล์ในเวอร์กสเตชันจะเป็นเพียงไฟล์ชั่วคราว

จากตัวอย่างข้างต้นนี้เราจะเห็นว่าหน้าที่ของไคล์เอ็นต์และเซอร์ฟเวอร์นั้นเปลี่ยนไปเพียงเล็กน้อยคือ ไคล์เอ็นต์ยังมีหน้าที่รับข้อมูลจากคีย์บอร์ด แปลรหัสข้อมูล และแสดงผลบนจอภาพ ส่วนเซอร์ฟเวอร์นั้นก็ยังมีหน้าที่เรียกหาข้อมูลจากจานแม่เหล็กให้แก่ไคล์เอ็นต์ แต่เรามาลดจำนวนครั้งของการเรียกหาข้อมูลลงไป งานจึงดูจะดีขึ้น

การลดภาระของไฟล์เซอร์ฟเวอร์นี้ยังทำได้อีกหลายรูปแบบ เช่น ในการเตรียมข้อมูลเพื่อใช้อัพเดตนั้นเราไม่จำเป็นต้องเก็บรายการอัพเดตไว้ที่ไฟล์เซอร์ฟเวอร์ เราสามารถเก็บไว้ที่เวอร์กสเตชัน เมื่อเตรียมข้อมูลเสร็จแล้วจึงเรียกโปรแกรมอัพเดตขึ้นมาทำงาน ด้วยวิธีนี้เราจะลดภาระของไฟล์เซอร์ฟเวอร์ได้หลายเท่าตัว ในการดึงข้อมูลออกมาทำรายงานก็เช่นกัน แทนที่เราจะดึงข้อมูลออกมาใส่เป็นไฟล์ไว้ในเซอร์ฟเวอร์ แต่เราสามารถดึงลงมาไว้ที่เวอร์กสเตชันเลยทีเดียว หลังจากนั้นแล้วก็สามารถออกรายงานได้ด้วยตัวเวอร์กสเตชันโดยตรง

อ่านมาถึงตรงนี้บางท่านอาจเริ่มคิดในใจว่าอาจารย์ปัญญานี่คิดอะไรตื้น ๆ และเริ่มจะผิดหวัง เรื่องนี้ต้องใจเย็น ๆ ครับ การโยกย้ายไฟล์บางชนิดออกมาจากไฟล์เซอร์ฟเวอร์นั้นเป็นเรื่องง่าย ๆ ก็จริง แต่มันชี้ถึงหลักการที่สำคัญอันหนึ่งคือ "งานอะไรที่ไม่จำเป็นต้องอยู่ในเซอร์ฟเวอร์ก็ต้องรีบย้ายไปอยู่ในหน้าที่ของไคล์เอ็นต์" สาเหตุที่ทำให้ผมแนะนำอย่างนี้ก็เพราะ จริง ๆ แล้วเซอร์ฟเวอร์และเวอร์กสเตชันนั้นต่างก็เป็นไมโครคอมพิวเตอร์ ความสามารถของมันจึงไม่แตกต่างกันนัก แต่เซอร์ฟเวอร์นั้นมีอยู่เพียงตัวเดียว มันต้องเซอร์ฟเจ้าเวอร์ก-

สเตชันนับ 10 นับ 100 ตัว มันจึงอาจเป็นตัวคอขวด (Bottle neck) ได้ ถ้าเราสามารถลดภาระของเซอร์ฟเวอร์ได้ก็จะทำให้ระบบทำงานได้มากขึ้น

จากหลักการดังกล่าวนี้เราสามารถนำมาประยุกต์เป็นเรื่อง ๆ ได้ คือเราลองมาคิดดูว่าโดยปกติแล้วโปรแกรมธรรมดา ๆ (Centralized) ต้องทำอะไรบ้าง เมื่อแยกเป็นฟังก์ชันหรือขั้นตอนได้แล้วก็มาพิจารณาว่าเราควรจะโยนให้ไคล์เอ็นต์หรือเซอร์ฟเวอร์ทำ

เรามาลองดูงานประเภทออนไลน์เรียลไทม์ และมัลติยูสเซอร์ เป็นตัวอย่าง งานแบบนี้เราต้องมีมาสเตอร์ไฟล์อยู่อันหนึ่งหรือชุดหนึ่ง เราต้องการอัพเดตจากเทอร์มินัลหรือเวอร์กสเตชันหลายตัว เทอร์มินัลเหล่านี้จะทำงานขนานกัน หรือคือ ต่างคนต่างทำ แต่แชร์มาสเตอร์ไฟล์ชุดเดียวกัน ในการอัพเดตนั้นแต่ละคนจะทำงานทีละรายการ หรือคือ ต้องเริ่มต้นด้วยการคีย์ข้อมูลที่จะต้องใช้เพื่ออัพเดตเข้ามา 1 รายการ เสร็จแล้วโปรแกรมจะต้องเช็คดูความเป็นไปได้ของข้อมูล (Editing) ถ้าผิดปกติก็จะบอกให้ผู้ใช้เทอร์มินัลทำการแก้ไข ถ้าไม่มีอะไรผิดปกติก็จะทำการอัพเดตแล้วแจ้งผลกลับไปที่จอภาพ ถ้าทุกอย่างเรียบร้อยดีก็เป็นอันว่าเสร็จรายการนั้น พนักงานก็จะทำรายการต่อไป เนื่องจากการอัพเดตนี้มีผลในทันที เราจึงเรียกมันว่าเป็นระบบ "เรียลไทม์" งานลักษณะนี้มีมากที่สุด เราจึงควรนำมาพิจารณาดู

ครับ จากที่บรรยายมา โปรแกรมของเราทำฟังก์ชันต่าง ๆ ดังนี้

(ก) แสดงภาพบนจอเพื่อการรับข้อมูล (Display)

(ข) รับข้อมูลจากคีย์บอร์ด (Input)

(ค) ตรวจดูความเป็นไปได้ของข้อมูล (Editing)

(ง) อ่านเอาเรคอร์ดที่ต้องการจากไฟล์ (Read)

(จ) ปรับแก้ข้อมูลในเรคอร์ด (Change/Update)

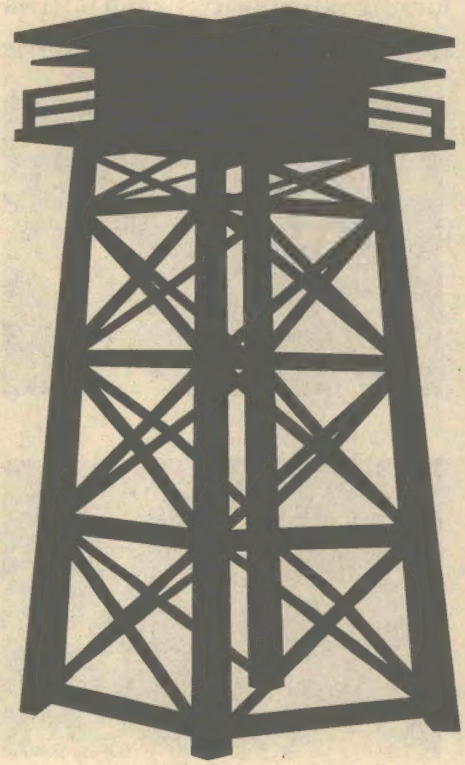
(ฉ) บันทึกเรคอร์ดนั้นคืนกลับไป (Write)

(ช) แสดงผลการอัพเดต (Result Display)

นี่เป็นการแบ่งอย่างคร่าว ๆ และดูแต่ในกรณีของการแก้ไขข้อมูลในเรคอร์ดที่มีอยู่แล้ว (Chenge) ผมคิดว่าเราคงจะขยายไปถึงกรณีอื่น ๆ ได้ด้วยตนเอง คำถามในตอนนี้ก็คือ ฟังก์ชันเหล่านี้ควรแบ่งให้ใครทำ ?

อธิบายฟังก์ชัน

ฟังก์ชันแรกนั้นผมคิดว่าทุกคนคงจะเห็นด้วยกับผม คือต้องเป็นหน้าที่ของเวอร์กสเตชัน เพราะมันอยู่ที่จอภาพของเวอร์กสเตชัน ฟังก์ชันที่ 2 นั้นก็เช่นเดียวกัน การกดแป้นที่เวอร์กสเตชันย่อมจะต้องรับโดยโปรแกรมในเวอร์กสเตชัน แต่พอมาถึงเรื่องการตรวจข้อมูล (Editing) เราอาจให้เป็นหน้าที่ของเวอร์กสเตชันหรือไฟล์เซอร์ฟเวอร์ก็ได้ ในระบบพีซีแลนของโนเวลล์นั้นเราสามารถเขียนโปรแกรมขึ้นมาเป็นโมดูล แล้วโหลดเข้าไว้ที่ตัวไฟล์เซอร์ฟเวอร์ได้ เขาเรียกมันว่า "NLM" ซึ่งย่อมาจากคำว่า "Netware Loadable Module" โมดูลนี้ทำหน้าที่ตรวจดูความเป็นไปได้ของข้อมูล ถ้าเราเลือกวิธีนี้ เราก็จะต้องให้โปรแกรมในเวอร์กสเตชันส่งข้อมูลให้กับโมดูลนี้ โมดูลนี้ตรวจเสร็จแล้วก็จะต้องส่งต่อไปยังโมดูลที่จะทำหน้าที่อ่านเอาเรคอร์ดขึ้นมา โมดูลที่ 2 นี้อาจทำฟังก์ชันในข้อ (ง), (จ), (ฉ) แล้วส่งผลกลับไปให้โปรแกรมในเวอร์กสเตชันได้ สำหรับฟังก์ชันสุดท้ายก็จะเป็นหน้าที่ของโปรแกรมในเวอร์กสเตชัน



ผมอยากให้เราพิจารณาฟังก์ชันในข้อ (ค) ให้ดี ๆ เรื่องนี้ถ้าโยนไปใส่ไว้ในไฟล์เซอร์ฟเวอร์มันก็จะมีข้อดีคือ มันเป็นโมดูลเดียว แต่เซอร์ฟได้ทุกเทอร์มินัล มันจึงประหยัดการใช้เมมโมรี ถ้าเราโยนให้เป็นหน้าที่ของโปรแกรมที่เวอร์กสเตชัน เราก็จะต้องมีโมดูลนี้อยู่ในทุกเวอร์กสเตชัน มันจึงกินเมมโมรีสูงขึ้น แต่การบรรจุไว้ในไฟล์เซอร์ฟเวอร์นั้นก็มีข้อเสีย คือมันกินกำลังเครื่องในไฟล์เซอร์ฟเวอร์ และถ้ามีข้อมูลส่งเข้ามาจากหลาย ๆ เทอร์มินัลพร้อม ๆ กัน มันจะต้องรอคิวกัน หรือคือเป็นภาระของซีพียูที่ทำหน้าที่เป็นไฟล์เซอร์ฟเวอร์มากขึ้น เรื่องนี้ผมแนะนำว่าควรกระจายอยู่ในเวอร์กสเตชัน ทั้งนี้เพราะเมื่อกระจายออกไปแล้วจะทำงานขนานกันได้ โปรแกรมในเวอร์กสเตชันต่าง ๆ นั้นไม่ต้องรอคอยซึ่งกันและกัน เมมโมรีในเวอร์กสเตชันนั้นมักจะมีจนเหลือเฟืออยู่แล้ว

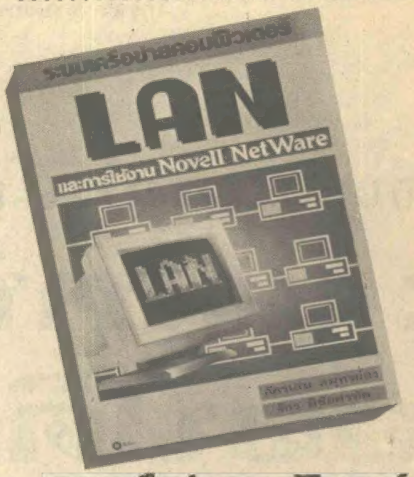
ด้วยหลักการเดียวกันนี้เราอาจบอกว่าฟังก์ชันในข้อ (จ) นั้นควรจะย้ายไปอยู่ในเวอร์กสเตชันด้วย ซึ่งถ้าทำอย่างนั้นเราก็จะพบว่างานที่เหลืออยู่ในไฟล์เซอร์ฟเวอร์ก็คือการอ่านและบันทึกข้อมูลลงจานแม่เหล็กเท่านั้น แต่เรื่องนี้อาจต้องคิดให้ละเอียดยิ่งขึ้น คือถ้าเราจับเอาฟังก์ชัน (ง), (จ) และ (ฉ) มาอยู่รวมกันที่ไฟล์เซอร์ฟเวอร์ เราจะพบว่าเราสามารถลดจำนวนครั้งของการรับส่งข้อมูลผ่านสายสื่อสารได้ ข้อนี้อาจจำเป็นก็ได้ เรื่องนี้จะขึ้นอยู่กับภาระ (Work load) ของสายสื่อสารเมื่อเทียบกับซีพียูของไฟล์เซอร์ฟเวอร์ ถ้าในระบบของเรามีการรับส่งข้อมูลระหว่างโปรแกรมในเวอร์กสเตชันกับในไฟล์เซอร์ฟเวอร์สูงมากอยู่แล้ว เราก็ควรจะโยนภาระในข้อ (จ) ไปอยู่ในไฟล์เซอร์ฟเวอร์ แต่ถ้ามีงานคำนวณมาก ๆ อยู่ในไฟล์เซอร์ฟเวอร์อยู่แล้ว เช่น กรณีที่เราใช้ดาต้าเบสอยู่ในไฟล์เซอร์ฟเวอร์ เราก็ควรจะโยนข้อ (จ) ไปอยู่ที่เวอร์กสเตชัน เรามองภาพอย่างนี้ได้แล้ว การตัดสินใจว่าจะให้โปรแกรมนั้นอยู่ที่ซีพียูตัวไหนก็ง่ายขึ้น และการควบคุมให้เขียนโปรแกรมได้ถูกต้องก็จะง่ายขึ้นด้วย

วันนี้ผมขอเสนอแนะแค่นี้ก่อน วันหลังจะพูดถึงหลักการข้ออื่น ๆ อีกสัก 2-3 ข้อ

สรุปส่งท้าย

ครับ นี่เป็นการคุยกันหรือแนะแนวเล็ก ๆ น้อย ๆ ให้แก่ท่านผู้อ่านเท่านั้น ผมคิดว่าเรื่องของการแยกฟังก์ชันและการแบ่งว่าจะให้ใครทำนี้เป็นเรื่องที่ต้องคิดเพิ่มเติมอีกมาก สิ่งที่ผมอยากให้ข้อสังเกตอีกอย่างหนึ่งคือ ฟังก์ชันต่าง ๆ ที่เราแยกออกมานี้อาจยกให้เป็นภาระของโปรแกรมแบบ 1 ต่อ 1 ก็ได้ หรือคือ เขียน 1 โปรแกรมต่อ 1 ฟังก์ชัน แล้วโปรแกรมเหล่านี้ก็จะทำงานเป็นลูกโซ่ตามลำดับไป แต่ละโปรแกรมก็จะทำหน้าที่เป็นทั้งไคล์-เอ็นต์และเซอร์ฟเวอร์อยู่ในตัว คือมันจะเป็นไคล์เอ็นต์เมื่อมันเป็นผู้ส่งข้อมูลให้คนอื่น และมันจะเป็นเซอร์ฟเวอร์เมื่อมันรับข้อมูลมาเพื่อดำเนินการตามหน้าที่ ถ้า

หนังสือดี ซีเอ็ด



ระบบเครือข่ายคอมพิวเตอร์

LAN

■ *โดย อัครเสน สมุทรผ่อง/จักร พิชัยศรทัต*
ราคาเล่มละ 220 บาท

LAN (Local Area Network) มีความสามารถที่เรียกได้ว่า เท่าเทียมกับเครื่องคอมพิวเตอร์ระดับมินิหรือเมนเฟรม ทั้งในด้านความเร็วในการทำงานและขนาดความจุข้อมูล ในขณะที่มีราคาถูกกว่า นอกจากนี้การหาบุคลากรเพื่อเรียนรู้และทำงานกับระบบ LAN ก็ทำได้ง่ายและรวดเร็วกว่าด้วย ดังนั้นหนังสือเล่มนี้จึงเหมาะสำหรับนักคอมพิวเตอร์และผู้สนใจทุกระดับ โดยเฉพาะผู้ที่มีความเกี่ยวข้องกับระบบ LAN

หาซื้อได้ตามร้านหนังสือทั่วไป
หรือสั่งซื้อโดยตรงได้ที่

บริษัท ซีเอ็ดยูเคชั่น จำกัด
800/43-45 ซอยตระกูลสุข ถนนอโศก-ดินแดง ห้วยขวาง กรุงเทพฯ 10400 โทร. 248-6280-9 (เวลาทำการ จันทร์-ศุกร์ 8.10-17.30 น.) โดยส่งธนาณัติสั่งจ่าย ปท.ดินแดง ในนามบริษัทฯ หรือเช็คส่วนตัวของธนาคารในเขตกรุงเทพฯ พร้อมค่าจัดส่ง

ค่าจัดส่ง สั่งซื้อไม่เกิน 100 บาท ค่าจัดส่ง 8 บาท สั่งซื้อเกิน 100 บาท ค่าจัดส่ง 8%

พิเศษ สำหรับสมาชิกวารสารของซีเอ็ด สั่งซื้อโดยตรงที่บริษัทฯ ได้ในราคาลด 15% เพิ่มค่าลงทะเบียน 4 บาท ค่าจัดส่งฟรี โดยระบุหมายเลขสมาชิกด้วย